

An Efficient Soft Decision Decoding Algorithm for Block Codes

H. J. Greenberger

Communications Systems Research Section

Nonoptimum decoding algorithms, which select a small set of candidate code words to be correlated with the received vector, can approach the performance of maximum likelihood decoders even at low signal-to-noise ratios. A decoding scheme based upon the best features of previously known algorithms of this type has been developed that can decode codes which have better performance than those in use today and yet not require an unreasonable amount of computation.

I. Introduction

An optimum decoder, for block codes transmitted over a memoryless channel, calculates the distance between the received vector and all possible code words and selects as the best estimate of the transmitted code word that one which is closest to it. Because of the amount of computation required, it is unreasonable to decode directly in this way all but the smallest codes, whose error correcting power is relatively weak. Many schemes have been suggested which reduce the computational complexity while maintaining the desired performance. The best of these algorithms are those which select a small subset of all possible code words, among which the best estimate of the full decoding algorithm has a high probability of being found. The selection of this subset uses the information provided by sorting the symbols of the received vector according to absolute magnitude so as to arrange them in order of their estimated probability of being correct. The algorithms of this class utilize this information and

the constraints of the parity check equations in different ways to generate the set of candidate code words.

One of the first algorithms of this kind was developed by D. Chase (Ref. 1). He suggested perturbing the hardlimited code word \mathbf{Y} by adding to it, modulo 2, a test pattern \mathbf{T} to obtain a new sequence \mathbf{Y}' . This new sequence is decoded algebraically to find the unique code word (if one exists) within half the minimum distance of the code. The nonzero bits of the test patterns are selected by a combinatorial construction on the least reliable bits of the received vector (Ref. 2).

L. Baumert, R. McEliece and G. Solomon (Refs. 3 and 4) have done much work using sets of erasure patterns. In their technique, a set of bits equal to the number of redundant bits in the code is erased. A candidate code word is then generated by reconstructing these bits from the unerased ones. Each erasure pattern generates another candidate to be correlated

with the received vector. The reasoning behind using erasure patterns rather than error patterns is that the redundancy of binary codes is much greater than its error correcting power. On the other hand, to correctly decode a received word, this scheme must cover all hard decision errors. (Using Chase's algorithm, a number of errors, up to the correcting power of the code, may remain exposed.)

These algorithms can be combined into a hybrid scheme which uses a small number of erasure masks, a few bits of redundancy and consideration of error patterns of weight two or less. Such an algorithm is computationally more efficient than any one of its ancestors and is a possible competitor, in terms of complexity vs performance, to Viterbi decoding of convolutional codes (Ref. 5).

II. An Efficient Hybrid Algorithm

This algorithm reduces the computational complexity by combining the best features of those mentioned in the previous section. Using the number of candidate code words required to achieve a given performance as a measure of efficiency, McEliece and Baumert's algorithm is the best. However, the amount of computation required to generate these candidates is much greater than for the other schemes. For each erasure mask the erased bits must be solved for in terms of the unerased ones. To reduce the number of times this computation must be done, errors of low weight are allowed in the unerased bits and redundancy is used to reduce the number of error patterns that need to be checked. A flow chart of the major sections of the algorithm is shown in Fig. 1.

The first step of the algorithm is to sort the symbols of the received code word according to their absolute magnitude, permute the columns of the parity check matrix and reduce it to standard form. Each erasure mask erases fewer bits than the number of parity check equations so not all of the unerased bits are independent. This dependency can be taken advantage of in order to determine which error patterns in the unerased bits are consistent with the parity equations, and only those patterns need be used to generate candidate code words. The number of such error patterns can be a small fraction of the total number of error patterns, greatly reducing the number of candidates required for a given level of performance.

Calculating the error patterns which will be consistent with the parity check equations can best be done by considering a portion of the syndrome and determining the error patterns, which when added to the initial estimate of the received vector will make that portion equal to zero. Since

a code word must satisfy $[H]c = 0$, it will also satisfy this equation for any subset of rows of $[H]$. Reducing the parity check matrix and partitioning $[H]$ as:

$$\begin{array}{cc} \left[\begin{array}{cc} P_1 & 0 \\ P_2 & 1 \end{array} \right] & \left[\begin{array}{c} c_1 \\ c_2 \end{array} \right] = 0 \\ r\text{-bits} & (n-r)\text{ bits} \end{array}$$

it is seen that $[P_1] c_1 = 0$. Therefore, the first r bits of the code word must also be the solution of a set of homogeneous equations.

For an arbitrary received vector a , the product $[H] a = s$ is called the syndrome and specifies the coset containing the possible error patterns in a . The same notion can be used when considering only the first r bits of the received word. Then $[P_1] a_1 = s_1$, where s_1 is the partial syndrome which specifies the possible error patterns in a_1 . Representing a_1 by $c_1 + e_1$, where e_1 is the partial error pattern corresponding to the partial code word c_1

$$[P_1] [c_1 + e_1] = s_1$$

or

$$[P_1] e_1 = s_1$$

Given $[P_1]$ and s_1 , there are a large number of partial error patterns e_1 that will satisfy this equation. However, for the decoding algorithm to be considered here, it is sufficient to consider patterns of 0, 1 or 2 errors. Note that even though the adjective "partial" is applied to a_1 , c_1 and s_1 , the remainder of the code word is completely determined from c_1 by $c_2 = [P_2] c_1$. The advantage of this approach is that candidate code words are determined only by possible error patterns in the most reliable received bits a_1 . The remaining received bits a_2 do not enter at all into the calculation and can be considered erasures.

No errors as a possible error pattern can only occur if the partial syndrome equals zero; single errors can occur in those bits whose corresponding columns of $[P_1]$ sum to s_1 . In general, for an error pattern of weight w to be a possibility, the sum of the w corresponding columns of $[P_1]$ must equal s_1 .

Efficient algorithms have been developed for all the steps of Fig. 1 and are described in detail in Ref. 6. As an example consider the rate 1/2, (128,64) BCH code of minimum distance 22. This code's maximum likelihood coding gain, at a

bit probability of error of 10^{-3} , is 1 dB better than the constraint length 7, rate 1/2 convolutional code which is in wide use today. The parameters for decoding this code using the algorithm developed here are:

- (1) Redundancy: 6 bits.
- (2) Number of masks: 20.

- (3) Mask weighing: see Fig. 2 (30 least reliable bits always erased).
- (4) Average Hamming distance between masks: 16.
- (5) All possible error patterns of weight 2 or less in un-erased bits checked.

The performance, using these parameters, is given in Fig. 3.

References

1. Chase, D., "A Class of Algorithms for Decoding Block Codes with Channel Measurement Information," *IEEE Trans. Inform. Theory*, Vol. IT-18, Jan. 1972, pp. 170-182.
2. Chase, D., and Goldfein, H. D., "Long Block Codes Can Offer Good Performance," Information Theory International Symposium, Cornell, N.Y., Oct. 1977.
3. Baumert, L., McEliece, R., and Solomon, G., "Decoding With Multipliers," in *The DSN Progress Report 42-34*, Jet Propulsion Laboratory, Pasadena, Calif., May-June 1976, pp. 43-46.
4. Baumert, L., and McEliece, R., "Soft Decision Decoding of Block Codes," *The DSN Progress Report 42-47*, Jet Propulsion Laboratory, Pasadena, Calif., July-August 1978, pp. 60-64.
5. Forney, G. D., Jr., "The Viterbi Algorithm" *Proc. IEEE*, Vol. 61, Mar. 1973, pp. 268-276.
6. Greenberger, H., Ph.D. Thesis, University of Southern California, 1978.

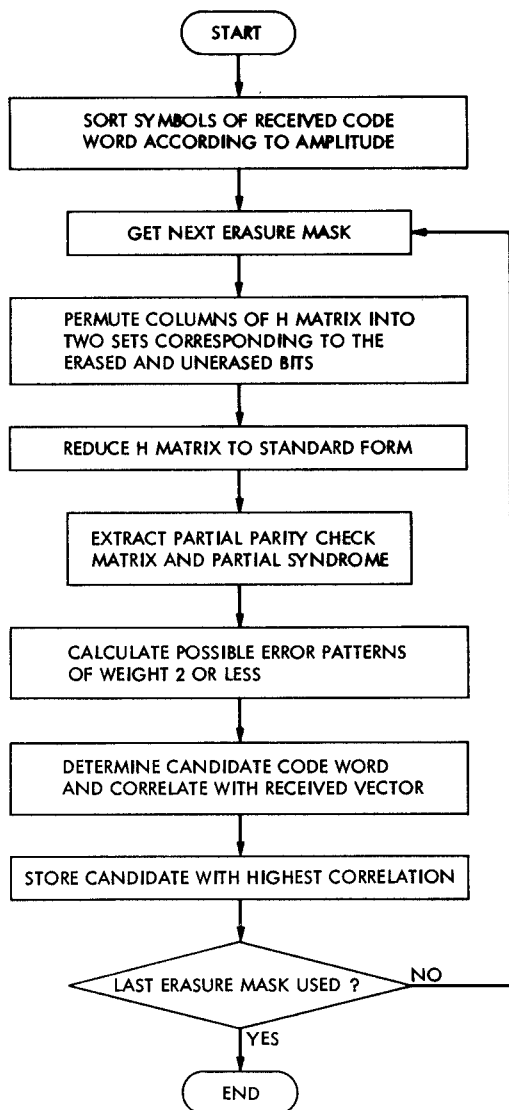


Fig. 1. Flow chart of soft decision decoding algorithm

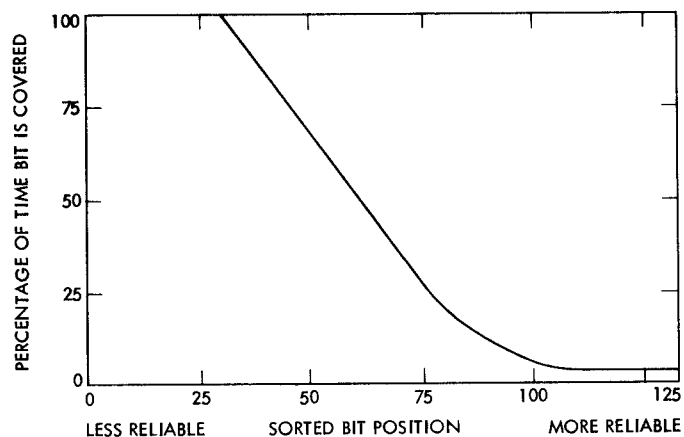


Fig. 2. Mask weighing function

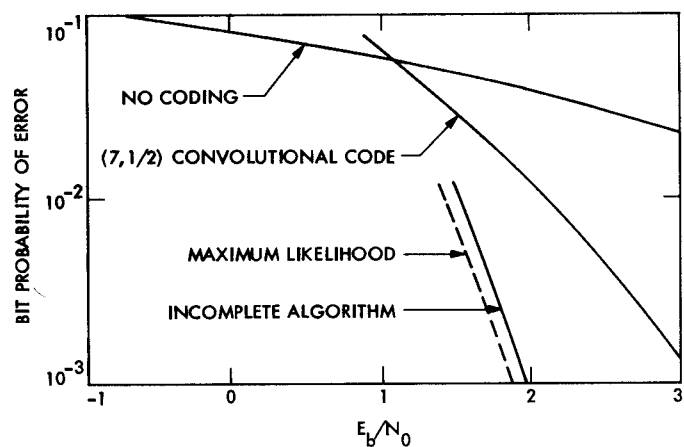


Fig. 3. Performance of the (128,64) BCH code